THE UNIVERSITY OF
MELBOURNE

Department of
Computer Science and Software Engineering

Software Requirements Specification

for

Managing the analysis of massive DNA sequence data

Version: 2.0
October 30, 2009

This document will specify what the client expects out of the system and also what the project team will deliver.

THE UNIVERSITY OF
MELBOURNE

Copyright notice

Department of Computer Science and Software Engineering
The University of Melbourne
Victoria
AUSTRALIA
3010

ICT Building
111 Barry Street
Carlton

Tel: +613 8344 1300
Fax: +613 9348 1184
https://www.csse.unimelb.edu.au/

Version: 2.0
October 30, 2009.
https://www.cs.mu.oz.au/SE-projects/s340gd

Credits

This document was written by members of Team D

CONTENTS

1 Introduction to the Document 1

1.1 Purpose of Software Requirements Specification 1

1.2 Description of the Document 1

1.3 Definitions, abbreviations and acronyms 2

1.4 Document Overview 2

2 Description of current and proposed system 3

2.1 Background Information 3

2.2 Existing system 3

2.3 Proposed System 3

2.4 User Characteristics 4

2.5 Assumptions and dependencies 4

3 Use Cases 6

3.1 Input File Select 6

3.2 Identify Short Read 7

3.3 Input Sequence's Identification Region Select 8

3.4 Input Sequence Region Select for BLAST 9

3.5 Group Input Sequence 10

3.6 Reference File Select 11

3.7 Reference File Region Select for BLAST 12

3.8 Output Column Select 13

3.9 Setting BLAST Parameter 14

4 Specific Requirements 15

4.1 Functional Requirements 15

4.2 Non-functional Requirements 25

5 System Boundaries and Constraints 27

6 Processing Flow 28

6.1 Flowchart 29

ONE

Introduction to the Document

1.1 Purpose of Software Requirements Specification

The purpose of the Software Requirements Specification is to formally specify what is wanted by the client and what the development team will be delivering. This document provides detailed explanations and reasons of each requirement of the system. The root causes of the current system and how the new system intends to solve these problems are explained in detail.

Throughout the development, the SRS will be used as a guide that the development team can refer to during the different stages of the software development cycle to ensure all the requirements have been met. The document will also be used in the future by the maintenance team at Peter MacCallum Cancer Centre to maintain the implemented system.

1.2 Description of the Document

This document will begin by explaining the purpose of the System Requirements Specification. A list of definitions, abbreviations and acronyms used throughout the document will be listed and explained. The references used in this document will also be identified. Furthermore a summary of the document will be provided.

Subsequently, a description of the current system will be in place. A description of the problems it causes will also be identified. Moreover, a description of the proposed system will be provided. An identification of the potential users of the new system will also be provided. The proposed system will be developed with assumptions and dependencies in mind. Hence, these assumptions and dependencies are also identified.

A list of requirements that the proposed system is to satisfy is also given. These requirements are separated into two modes. The first mode being the requirements that must be satisfied in order for the system to be considered complete. The second mode contains requirements that are optional, but desired. These requirements will be met if the development team has time to implement them. Lastly, a list of constraints to the proposed system is provided at the end.

1.3 Definitions, abbreviations and acronyms

ASCII Abbreviated for American Standard Code for Information Interchange. We will use two methods to determine the quality of a short read based on decimal ASCII values.

BLAST An algorithm for comparing primary biological sequence information, such as the the nucleotides of DNA sequences. There exists a program by the same name which performs a search based on the BLAST algorithm. Any references to BLAST in this document is in fact referring to the program.

CSV Stands for Comma-separated Values. Files encoded in this format have each item seperated by a comma, and rows seperated by new lines.

Development team A team consisting of Weng Hoe Ng, Jinita Patel, Jack Low and Lawrence Bang.

FASTA format (a.k.a. Pearson format) A text-based format for representing nucleotide sequences, in which base pairs or amino acids are represented using single-letter codes.

Input files (a.k.a. query files) Text files separated by colons to be converted to FASTA format where it may be used by BLAST to query a database.

Reference files CSV files which are used by BLAST as a reference to the input files. They must first be converted to a format which can be used by BLAST.

SRS Stands for Software Requirements Specification. It is a document where the requirements of our software that are planned to be delivered are listed.

The client The Peter MacCallum Cancer Centre and Jason Li.

1.4 Document Overview

This section gives a brief overview of the rest of the document.

- *Chapter I: Introduction*
  An introduction to the SRS; who is reading it, why it is important, and a list of common definitions used throughout the entire document.

- *Chapter II: Overall Description of the Proposed System*
  Descriptions of the current system and their problems, followed by the proposed system and solutions.

- *Chapter III: Use Cases*
  This section includes the use cases that will explain several cases of user interaction with the system.

- *Chapter IV: Specific Requirements*
  The solutions to the problems are listed in detail, with their purpose and technical data defined clearly.

Description of current and proposed system

2.1 Background Information

One of the research activities conducted by the biologists of Peter MacCallum Cancer Centre is to regularly compare short fragments of sampled DNA sequences to an existing library of known DNA sequences. This allows biologists to study genetic activities that have occurred in the sampled individual. BLAST is used to carry out the sequence comparison process. Prior to applying BLAST, data is trimmed as desired to remove junk data and also to be manipulated so as to extract the regions of interest that have occurred in the sampled individual. One type of regions of interest is the unique identifier that was attached to the sequences during the experiment and the other type is the region comparable to the library of known DNA sequences. A piece of software has been requested to be built in order to simplify their job and to reduce operational costs.

2.2 Existing system

The current system that Jason uses is not very effective nor efficient. R scripting is used to pre-process data so it may be used by BLAST and executed separately. This causes the mapping process to take several hours to generate results, which are only shown after the whole process has been completed. The results indicate the frequency (test.freq), the identification of the sample (oligo.ID), the gene (full hairpin sequence) and information of the gene (gene.info). These are represented in a table as results. Most biologists are also unable to carry out this process of grouping and selecting DNA sequences on their own as they do not have the computer knowledge and the experience. Hence queries are passed to a bioinformatician to be processed before being run through the BLAST. This causes one person to be responsible for all the queries that are sent by the biologists.

2.3 Proposed System

As mentioned above, Jason uses R scripting to pre-process data which is fed to BLAST to compare short-fragments of DNA sequences. Currently, the biologists do not have sufficient computer knowledge to do this on their own. Hence the development team will build a Graphical User interface that wraps around BLAST, which will enable biologists to pre-process data and BLAST on their own personal computers. The development team aims to create a piece of software that is easily learnable and is user friendly. As the biologists will be responsible for their own individual queries, it will solve the problem of one man being responsible for all the queries of the biologists. The new proposed system aims to speed up the process of BLASTing short-fragments of DNA sequences using multi-threading. Multiple threads of data will be BLASTed simultaneously to reduce the number of hours to generate results. In the future Jason is expecting more new data from external institutes. Hence the amount of data they have to work with will eventually increase. This piece of software is meant to simplify their work in the future.

2.4 User Characteristics

The main users of the system will be:

- biologists
- statisticians
- clinicians
- bioinformaticians

These are scientists at Peter MacCallum Cancer Centre that will be using the system to assist them in their research. These users will have sufficient knowledge in using the new system, as explained by Jason.

2.5 Assumptions and dependencies

Assumptions and dependencies made by development team about the program are documented below. If a bug exists in the future, the maintenance team at PMCC will know about it. And if a bug occurs in the future due to a changed format, the assumptions and dependencies are documented so the maintenance team are aware of them.

2.5.1 All data used by the system can be assumed to be correct and complete.

This assumption means that development team will have to check for erroneous files, as it will be hard to decide what is correct and what isn't.

2.5.2 The user will know whether an input sequence file has a paired end or not.

The system can not possibly know whether an input file has a paired end or not, so it is up to the user to know if there is a paired end file or not.

2.5.3 The reference file will have a header row.

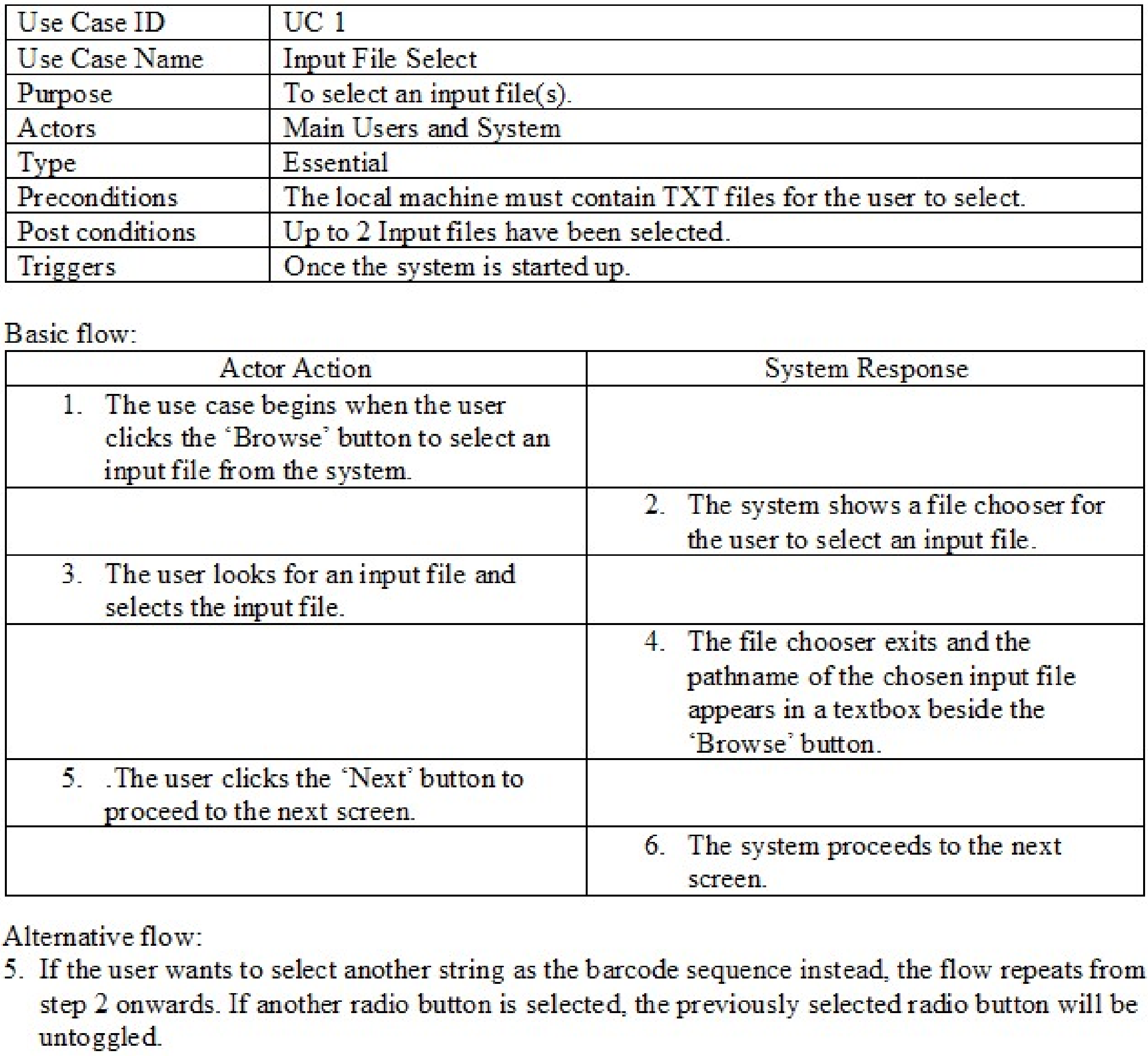
It is important that this assumption is correct or at the very least, consistent. It is impossible for the system to distinguish between files that have header rows and those that don't. This assumption will affect the way the system parses the reference files.

THREE

Use Cases

This section includes the use cases that will explain several cases of user interaction with the system.

3.1 Input File Select



Use Case ID	UC 1
Use Case Name	Input File Select
Purpose	To select an input file(s).
Actors	Main Users and System
Type	Essential
Preconditions	The local machine must contain TXT files for the user to select.
Post conditions	Up to 2 Input files have been selected.
Triggers	Once the system is started up.

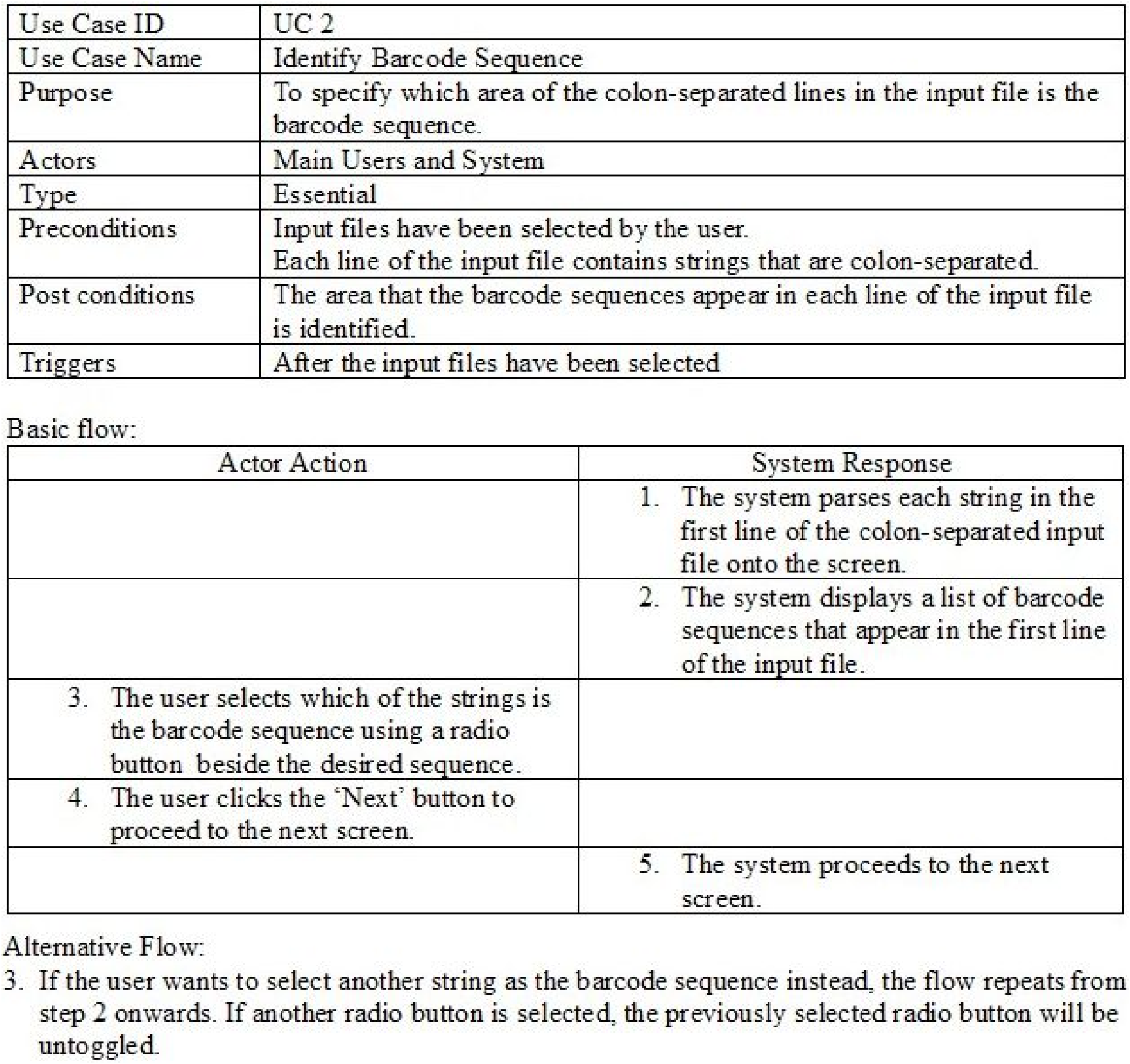
Basic flow:

Actor Action	System Response
1. The use case begins when the user clicks the 'Browse' button to select an input file from the system.	
	2. The system shows a file chooser for the user to select an input file.
3. The user looks for an input file and selects the input file.	
	4. The file chooser exits and the pathname of the chosen input file appears in a textbox beside the 'Browse' button.
5. .The user clicks the 'Next' button to proceed to the next screen.	
	6. The system proceeds to the next screen.

Alternative flow:

5. If the user wants to select another string as the barcode sequence instead, the flow repeats from step 2 onwards. If another radio button is selected, the previously selected radio button will be untoggled.

3.2 Identify Short Read



Use Case ID	UC 2
Use Case Name	Identify Barcode Sequence
Purpose	To specify which area of the colon-separated lines in the input file is the barcode sequence.
Actors	Main Users and System
Type	Essential
Preconditions	Input files have been selected by the user. Each line of the input file contains strings that are colon-separated.
Post conditions	The area that the barcode sequences appear in each line of the input file is identified.
Triggers	After the input files have been selected

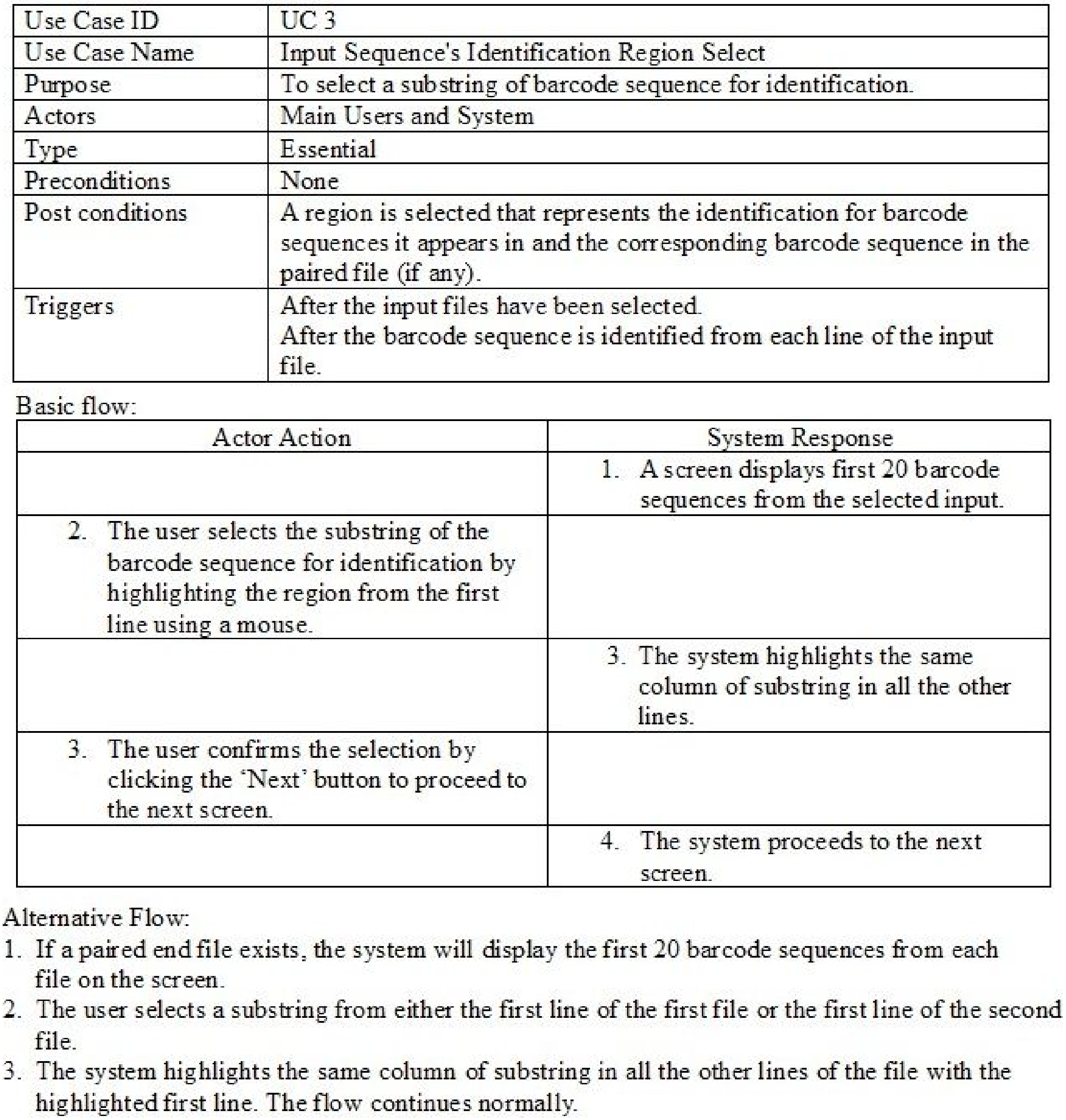
Basic flow:

Actor Action	System Response
	1. The system parses each string in the first line of the colon-separated input file onto the screen.
	2. The system displays a list of barcode sequences that appear in the first line of the input file.
3. The user selects which of the strings is the barcode sequence using a radio button beside the desired sequence.	
4. The user clicks the 'Next' button to proceed to the next screen.	
	5. The system proceeds to the next screen.

Alternative Flow:

3. If the user wants to select another string as the barcode sequence instead, the flow repeats from step 2 onwards. If another radio button is selected, the previously selected radio button will be untoggled.

3.3 Input Sequence's Identification Region Select



Use Case ID	UC 3
Use Case Name	Input Sequence's Identification Region Select
Purpose	To select a substring of barcode sequence for identification.
Actors	Main Users and System
Type	Essential
Preconditions	None
Post conditions	A region is selected that represents the identification for barcode sequences it appears in and the corresponding barcode sequence in the paired file (if any).
Triggers	After the input files have been selected. After the barcode sequence is identified from each line of the input file.

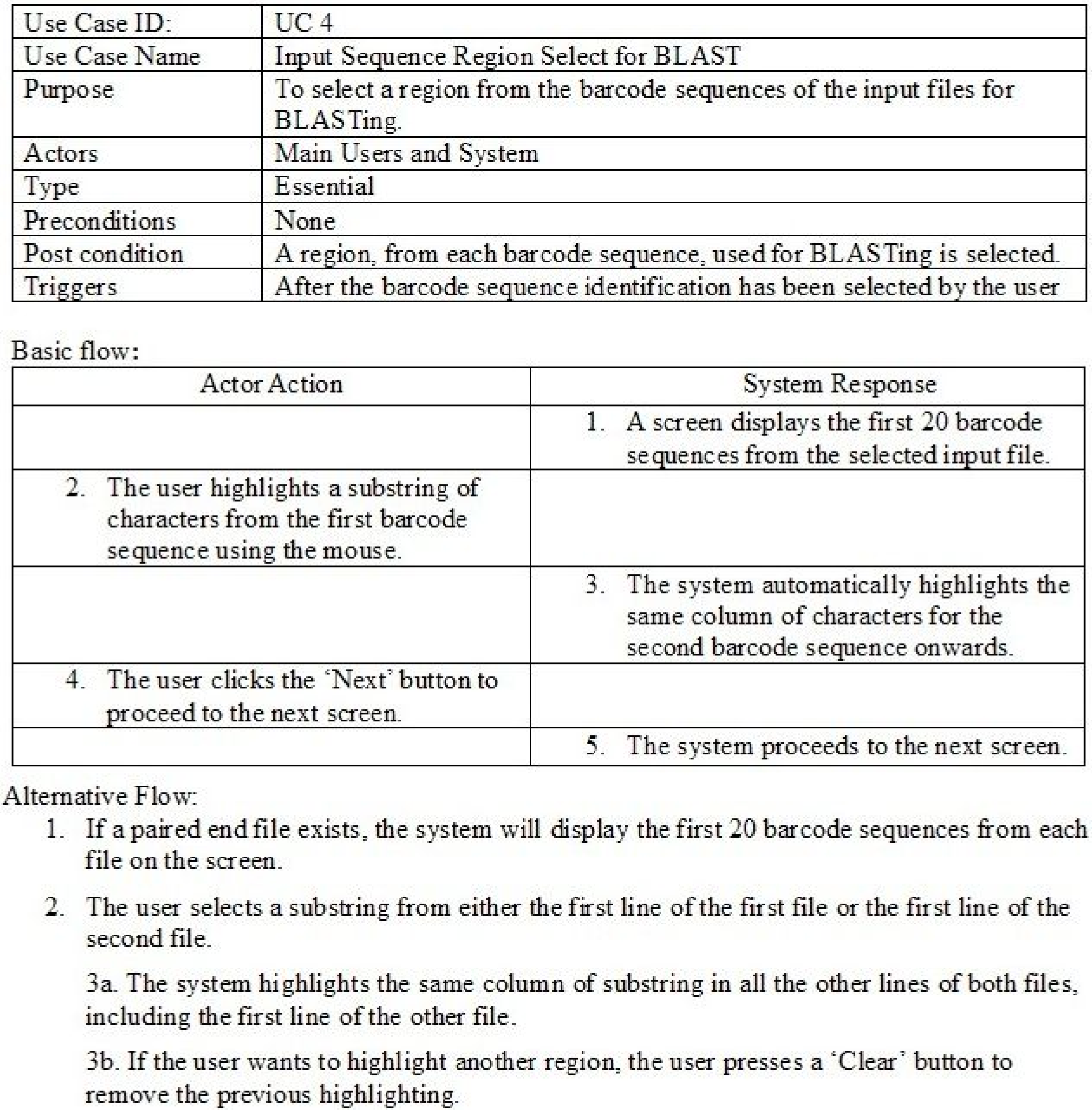
Basic flow:

Actor Action	System Response
	1. A screen displays first 20 barcode sequences from the selected input.
2. The user selects the substring of the barcode sequence for identification by highlighting the region from the first line using a mouse.	
	3. The system highlights the same column of substring in all the other lines.
3. The user confirms the selection by clicking the 'Next' button to proceed to the next screen.	
	4. The system proceeds to the next screen.

Alternative Flow:

1. If a paired end file exists, the system will display the first 20 barcode sequences from each file on the screen.
2. The user selects a substring from either the first line of the first file or the first line of the second file.
3. The system highlights the same column of substring in all the other lines of the file with the highlighted first line. The flow continues normally.

3.4 Input Sequence Region Select for BLAST



Use Case ID:	UC 4
Use Case Name	Input Sequence Region Select for BLAST
Purpose	To select a region from the barcode sequences of the input files for BLASTing.
Actors	Main Users and System
Type	Essential
Preconditions	None
Post condition	A region, from each barcode sequence, used for BLASTing is selected.
Triggers	After the barcode sequence identification has been selected by the user

Basic flow:

Actor Action	System Response
	1. A screen displays the first 20 barcode sequences from the selected input file.
2. The user highlights a substring of characters from the first barcode sequence using the mouse.	
	3. The system automatically highlights the same column of characters for the second barcode sequence onwards.
4. The user clicks the 'Next' button to proceed to the next screen.	
	5. The system proceeds to the next screen.

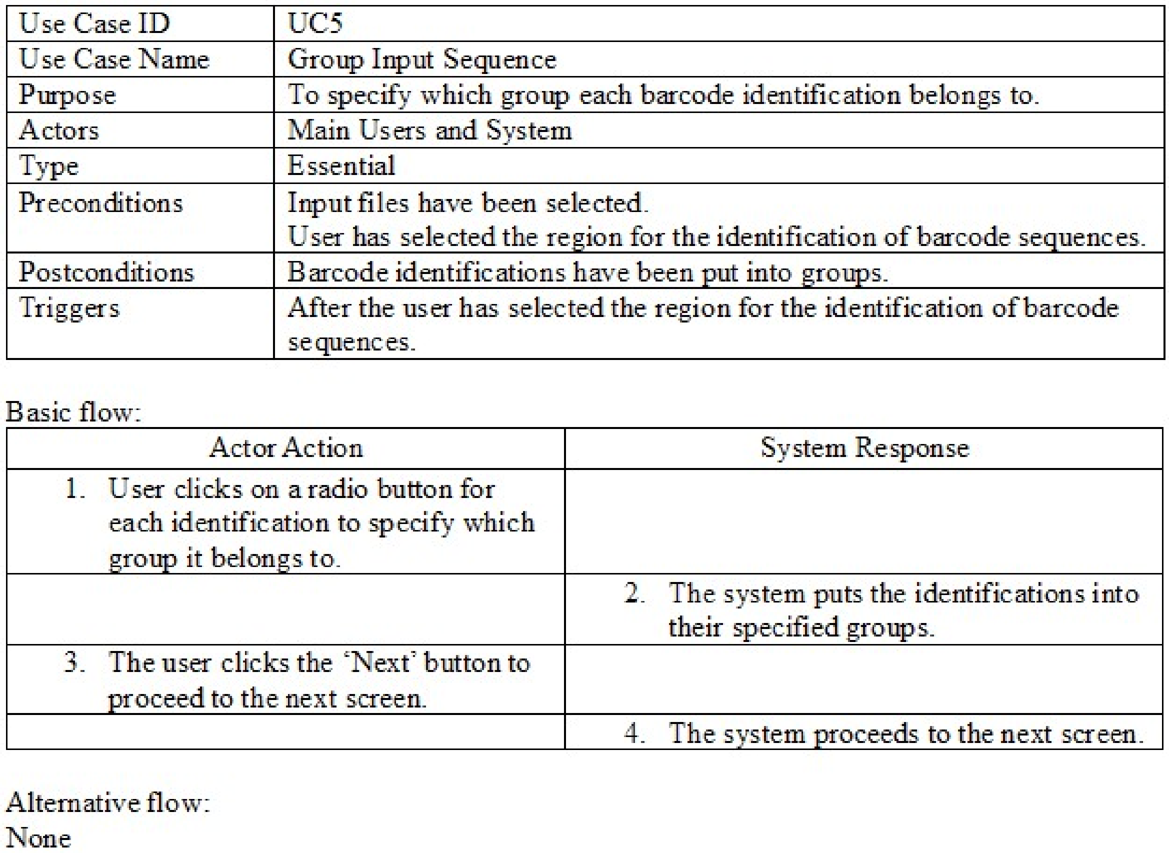
Alternative Flow:

1. If a paired end file exists, the system will display the first 20 barcode sequences from each file on the screen.
2. The user selects a substring from either the first line of the first file or the first line of the second file.

   3a. The system highlights the same column of substring in all the other lines of both files, including the first line of the other file.

   3b. If the user wants to highlight another region, the user presses a 'Clear' button to remove the previous highlighting.

3.5 Group Input Sequence



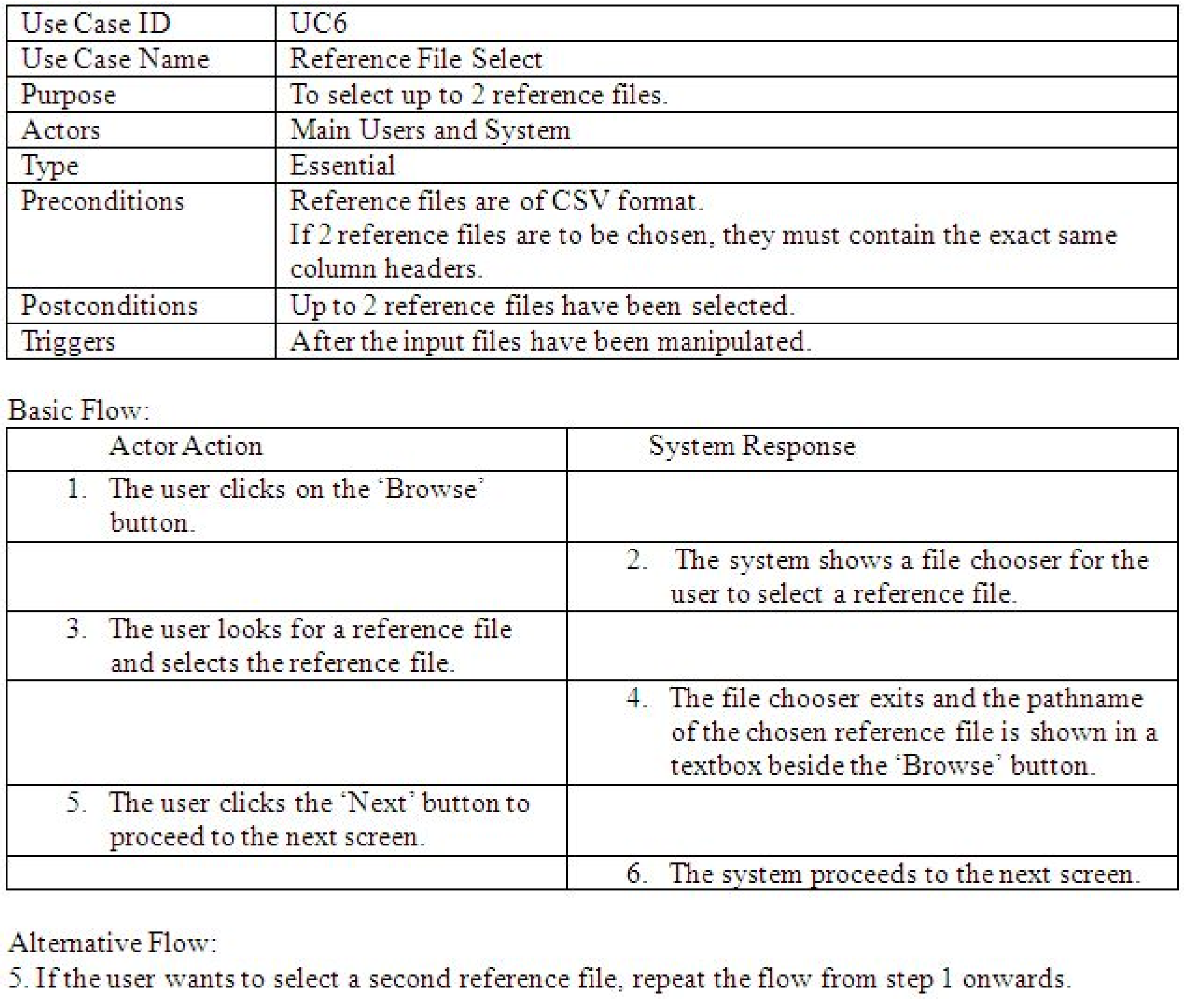
Use Case ID	UC5
Use Case Name	Group Input Sequence
Purpose	To specify which group each barcode identification belongs to.
Actors	Main Users and System
Type	Essential
Preconditions	Input files have been selected. User has selected the region for the identification of barcode sequences.
Postconditions	Barcode identifications have been put into groups.
Triggers	After the user has selected the region for the identification of barcode sequences.

Basic flow:

Actor Action	System Response
1. User clicks on a radio button for each identification to specify which group it belongs to.	
	2. The system puts the identifications into their specified groups.
3. The user clicks the 'Next' button to proceed to the next screen.	
	4. The system proceeds to the next screen.

Alternative flow:
None

3.6 Reference File Select



Use Case ID	UC6
Use Case Name	Reference File Select
Purpose	To select up to 2 reference files.
Actors	Main Users and System
Type	Essential
Preconditions	Reference files are of CSV format. If 2 reference files are to be chosen, they must contain the exact same column headers.
Postconditions	Up to 2 reference files have been selected.
Triggers	After the input files have been manipulated.

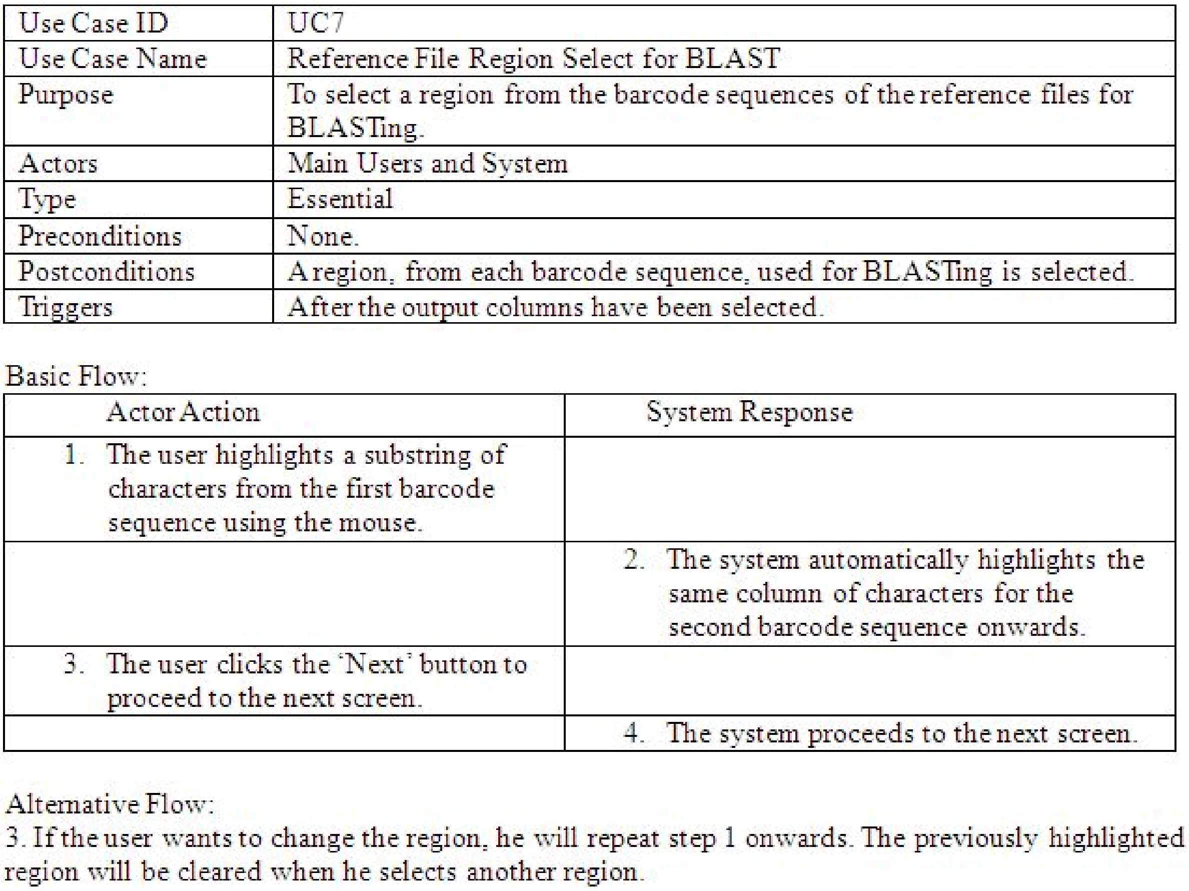
Basic Flow:

Actor Action	System Response
1. The user clicks on the 'Browse' button.	
	2. The system shows a file chooser for the user to select a reference file.
3. The user looks for a reference file and selects the reference file.	
	4. The file chooser exits and the pathname of the chosen reference file is shown in a textbox beside the 'Browse' button.
5. The user clicks the 'Next' button to proceed to the next screen.	
	6. The system proceeds to the next screen.

Alternative Flow:

5. If the user wants to select a second reference file, repeat the flow from step 1 onwards.

3.7 Reference File Region Select for BLAST



Use Case ID	UC7
Use Case Name	Reference File Region Select for BLAST
Purpose	To select a region from the barcode sequences of the reference files for BLASTing.
Actors	Main Users and System
Type	Essential
Preconditions	None.
Postconditions	A region, from each barcode sequence, used for BLASTing is selected.
Triggers	After the output columns have been selected.

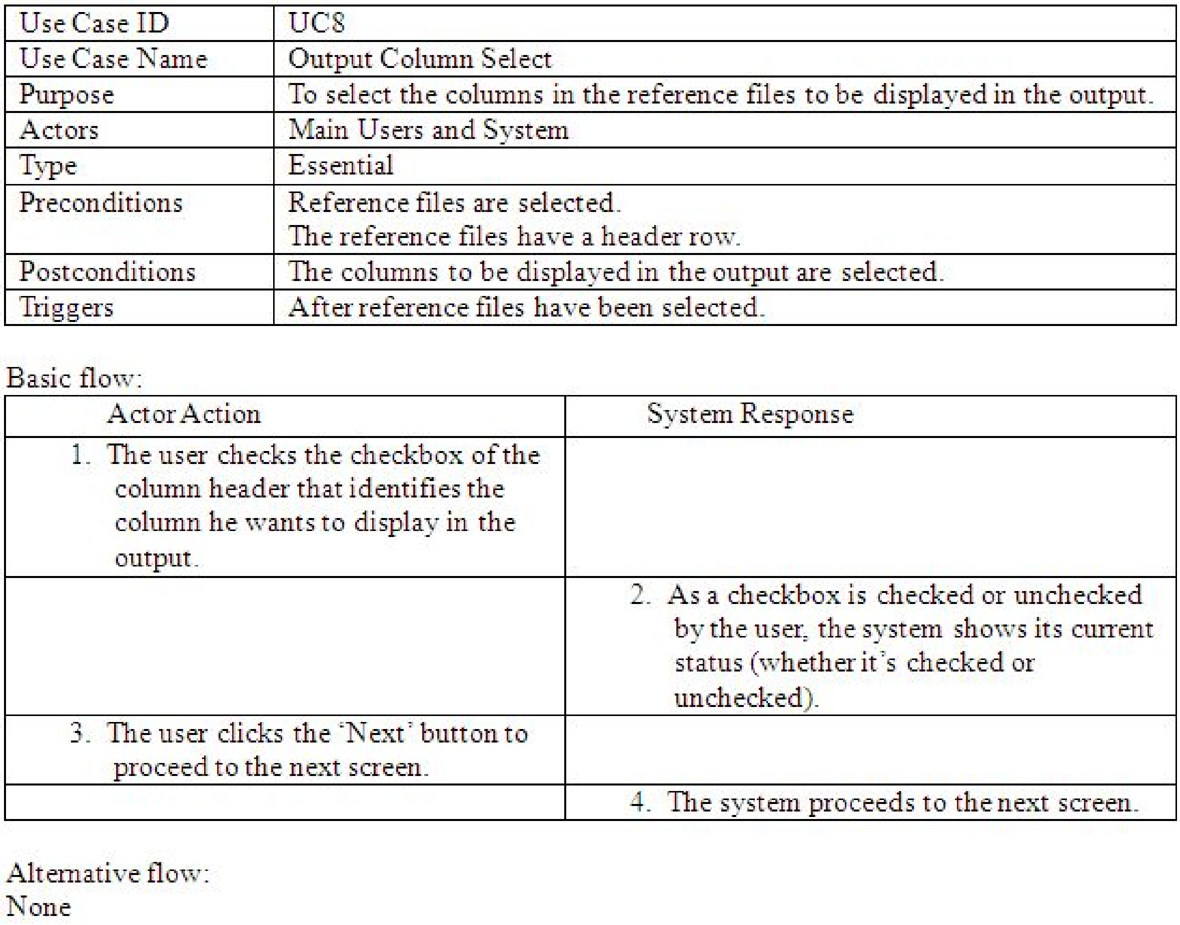
Basic Flow:

Actor Action	System Response
1. The user highlights a substring of characters from the first barcode sequence using the mouse.	
	2. The system automatically highlights the same column of characters for the second barcode sequence onwards.
3. The user clicks the 'Next' button to proceed to the next screen.	
	4. The system proceeds to the next screen.

Alternative Flow:

3. If the user wants to change the region, he will repeat step 1 onwards. The previously highlighted region will be cleared when he selects another region.

3.8 Output Column Select



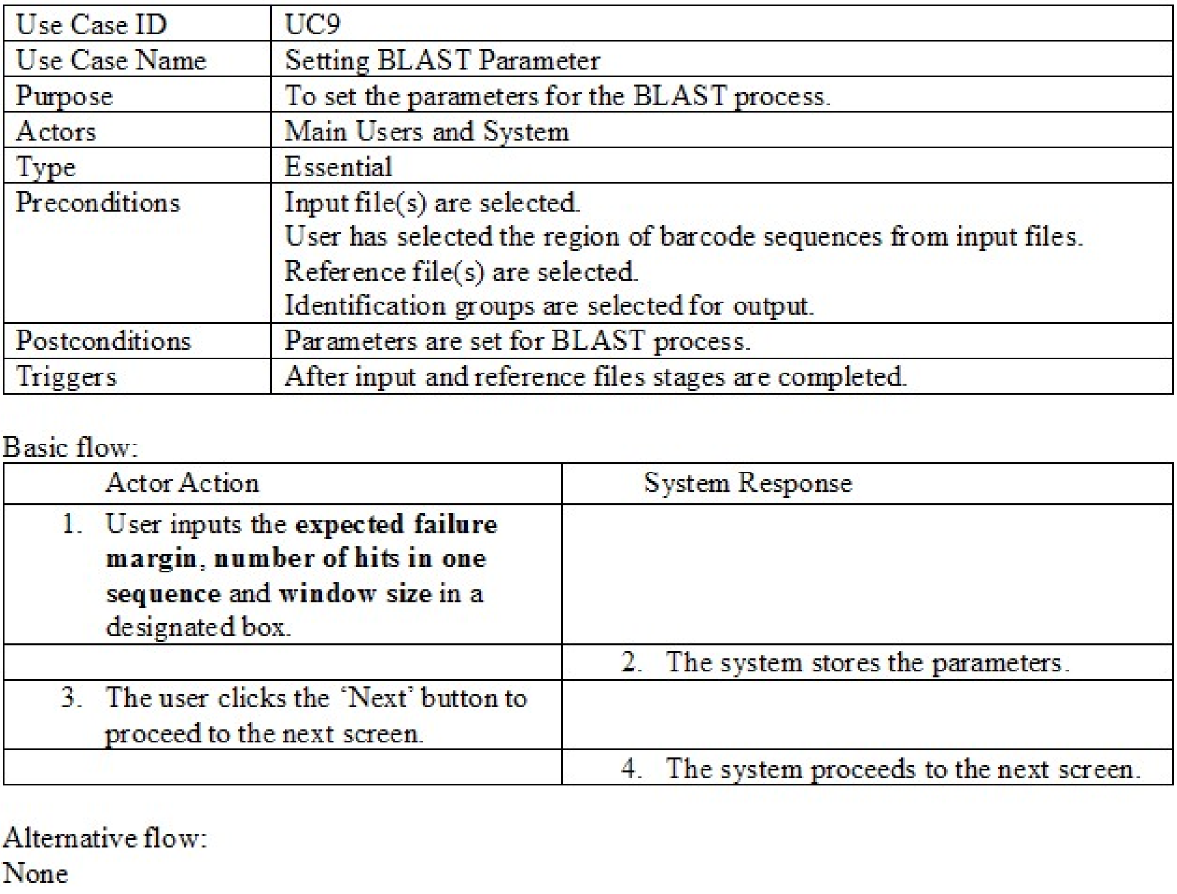
Use Case ID	UC8
Use Case Name	Output Column Select
Purpose	To select the columns in the reference files to be displayed in the output.
Actors	Main Users and System
Type	Essential
Preconditions	Reference files are selected. The reference files have a header row.
Postconditions	The columns to be displayed in the output are selected.
Triggers	After reference files have been selected.

Basic flow:

Actor Action	System Response
1. The user checks the checkbox of the column header that identifies the column he wants to display in the output.	
	2. As a checkbox is checked or unchecked by the user, the system shows its current status (whether it's checked or unchecked).
3. The user clicks the 'Next' button to proceed to the next screen.	
	4. The system proceeds to the next screen.

Alternative flow:
None

3.9 Setting BLAST Parameter



Use Case ID	UC9
Use Case Name	Setting BLAST Parameter
Purpose	To set the parameters for the BLAST process.
Actors	Main Users and System
Type	Essential
Preconditions	Input file(s) are selected. User has selected the region of barcode sequences from input files. Reference file(s) are selected. Identification groups are selected for output.
Postconditions	Parameters are set for BLAST process.
Triggers	After input and reference files stages are completed.

Basic flow:

Actor Action	System Response
1. User inputs the **expected failure margin**, **number of hits in one sequence** and **window size** in a designated box.	
	2. The system stores the parameters.
3. The user clicks the 'Next' button to proceed to the next screen.	
	4. The system proceeds to the next screen.

Alternative flow:
None

FOUR

Specific Requirements

4.1 Functional Requirements

This section documents functions the system must perform. These functions define what the system will accomplish.

Pre-processing Flow

4.1.1 Essential

This section will contain essential functional requirements for the pre-processing flow that must be present in the system in order for it to be considered complete. These requirements are of high priority and must be implemented by all means.

Input Files Stage

4.1.1.1 Input files selection

Description: The user must be able to select up to two input files for pre-processing. If a second input file is necessary, the user must specify that a paired end file exists using a checkbox before it is possible to browse for a second input file. The second input file is assumed to be a paired end file as of Assumption 2.4.3.

- The system must be able to let the user select an input file using a file chooser.

- The system must restrict file selection to only .txt files.

- There will be a ’Browse’ button to allow the user to open up a file chooser for selecting the first input file.

- If a paired end file exists, the user will specify it by checking a checkbox below the ’Browse’ button.

- The system must then allow the user to select a second input file using a ’Browse’ button that opens up a file chooser.

- The pathname of the selected files will be displayed in a textbox beside the corresponding 'Browse' button.

4.1.1.2 Default selection of short read from input file(s)

Description: The system will automatically specify the 6th colon separated string in each line of the input file(s) to be the short read. In the following example line, SIX represents the short read of the line.

```
ONE:TWO:THREE:FOUR:FIVE:SIX:SEVEN
```

4.1.1.3 Default selection of quality from input file(s)

Description: The system will automatically specify the 7th colon separated string in each line of the input file(s) to be the quality.In the following example line, SEVEN represents the quality of the line.

```
ONE:TWO:THREE:FOUR:FIVE:SIX:SEVEN
```

Quality Processing Stage

4.1.1.4 Specification of quality threshold for each input file

Description: The user will be able to specify a quality threshold, where sequences below a certain threshold will be filtered out. The two measures of quality will be the minimum ASCII value, and the second being the sum of the ASCII values. Using its character-encoding scheme, the English alphabets are converted to numerical values before the sum is obtained. These two measures are optional to use and either one or both measures can be used. The default values will be 60 and a number proportional to the length of the string respectively.

- The user will be able to specify a quality threshold for each input file if desired.

- If a short read's quality is below the specified threshold for the file it exists in, the line the short read is in will be filtered out.

4.1.1.5 Saving input files that have been filtered with sufficient quality

Description: The user will be able to save each of the input files after they have gone through quality processing. These new files will contain lines of short reads that are of sufficient quality. In comparison with the original input files, the quality processed input files will not have lines of "junk" and short reads of poor quality.

- The user will be able to save each of the input files after they have gone through quality processing.

- The system will generate these files as .txt files with only short reads of sufficient quality.

- The newly generated .txt files will be of the same format as the original .txt files (colon separated).

Grouping Stage

4.1.1.6 Region selection for each short read's identification

Description: If an input file has a paired end file, the user should be able to select which one of the pair he wants to select a region from for grouping purposes. If the user selects a region in the first file, then that region will represent the identification for the short read it exists in, and also represent the identification for the corresponding short read in its paired file.
If an input file has no paired end file, the user will select a region to represent the identification for each short read, similarly.

If paired end exists:
- The user will get to specify which columns of the first file or the second (paired end) file represents the identification, using the mouse.

- The system will save the selected columns for each line of the file as identification for the short read in that file and also the identification for its paired end in the other file.

If paired end does not exist:
- The user will get to specify which columns of the first file represents the identification, using the mouse.

- The system will save the selected columns for each line of the file as identification for the short read in that file.

4.1.1.7 Wildcards

Description: In order to speed up the grouping process, instead of having users manually select which identifiers belong in which group, the user should be able to utilise wildcards in identifiers. The '* wildcard will expand to one character. So, AC*T will match: ACGT, ACCT and so on.

- The system may allow users to use wildcards in identifiers.

- The system will interpret the '*' wildcard.

4.1.1.8 Grouping of input sequences

Description: Continuing from requirement 4.1.1.7, users will be able to select groups each identification belongs to. There will a minimum of two groups, and each identification can only appear in one group. For every group of short reads, there will be a file generated locally that contains every short read that exists in that group. In the case of paired-end input files, the corresponding line from the second file will also be placed in the same group and the files created must reflect this. These files will aid with the creation of the FASTA files.
If more groups are needed, the user will be able to add more groups.

- The system must allow the user to specify which group an identification belongs to by typing out the identifications in each corresponding textbox.

- An identification can only exist in one group.

- If the user wants more groups, the user must be able to add more groups using a button.

- The number of generated files containing a list of short reads for each group will be the same as the number of groups of short reads multiplied by the number of input files previously selected.

4.1.1.9 Saving grouped files that have been generated from grouping

Description: One file will be generated for each group of sequences from the grouping process. The user will be able to save each of these files onto the local machine.

- The user will be able to save each of the input files after they have gone through the grouping process.

- The system will generate these files as .txt files.

- The newly generated .txt files will be of the same format as the original .txt files (colon separated).

4.1.1.10 Tag file generation

Description: A tag file must be generated after the user has completed the pre-processing. A tag file must contain a set of unique short read and the frequency of the sequence in the input file. This will be generated by the system by counting the times a particular sequence appears in the input file.

- The system will generate the tag file after the user completes the input file stage.

- The system will allow the user to view the tag file that is generated.

4.1.2 Optional

This section documents the functional requirements for the pre-processing flow that are desired but considered optional by Jason. These requirements are understood to be useful to the end users, but are of lower priority compared to the essential requirements. They will only be implemented after all the essential requirements are done and if the development team has time to do so.

Input Files Stage

4.1.2.1 Check for conflicting identifications while grouping

Description: If the user has specified an identification to be in more than one group, whether it be manually or using a wildcard, the system will alert the user of this error.

- A message box will popup with a message to tell the user of a conflict.
- The user will have to fix the conflict before moving away from the grouping stage.

BLAST Processing Flow

4.1.3 Essential

This section will contain essential functional requirements for the BLAST processing flow that must be present in the system in order for it to be considered complete. These requirement are of high priority and must be implemented by all means.

Input File Stage

4.1.3.1 Input file selection

Description: The user must be able to select one input files for the BLAST process.

- The system must be able to let the user select an input file using a file chooser.

- The system must restrict file selection to only .txt files.

- There will be a button to allow the user to open up a file chooser for selecting the first input file.

- The pathname of the selected file will be displayed in a textbox beside the button.

4.1.3.2 Default selection of short read from input file

Description: The system will automatically specify the 6th column of each colon separated string in each line of the input file to be the short read. Please refer to Requirement 4.1.1.2 for an example.

4.1.3.3 Region selection for BLASTing input sequences

Description: Following on from Requirement 4.1.3.1, the user must be able to specify a region of short read from the input file to be used in the BLAST process. If the input file has a pair end, only one of the files will be used to select the region. The corresponding region in the paired end file will also be used for the BLAST process.

- The user will specify a region of short read from an input file to be used in the BLAST process.

- If a paired end file was previously chosen, the user will only specify a region from one of the two input files.

- The system will save the specified region to be used for the BLAST process.

- If a paired end file was previously chosen, the system will also save the corresponding region from the paired end file to be used for the BLAST process.

Reference Files Stage

4.1.3.4 Reference files selection

Description: A reference file is a file that contains library of DNA sequences. Reference files exist in the local machines and the user should be able to select the desired file from the disk. The user must be able to select multiple reference files. If more reference files are to be selected, a button may be pressed to allow more reference files to be selected Only CSV files may be selected.

- The system must be able to let the user select one reference file using a file chooser.

- The system must restrict file selection to CSV files.

- There will be a button to allow the user to open up a file chooser for selecting the file.

- The pathname of the selected file will be displayed in a textbox beside the corresponding 'Browse' button.

- A button may be pressed to allow the selection of another reference file.

- The system will allow up to five reference files to be chosen.

4.1.3.5 Output column selection

Description: After requirement 4.1.3.4 has been carried out, the user must be able to select columns that are present in the selected reference files to appear in the output file. The data that appears in the output table must be of the columns the user has selected. Reference files that are selected to BLAST from may contain different columns, hence all options must be presented to the user.

- The system will display all the column headers on the screen, with a checkbox beside each header.

- The user will check the checkboxes that correspond to the columns to be displayed in the output table.

4.1.3.6 Region selection for BLASTing reference sequences

Description: Following on from requirement 4.1.3.5, the user must be able to select a substring of short read, from the original hairpin sequence from the reference file(s), used for BLASTing. A default number of short reads displayed will be determined by the system as mentioned by Jason. The user has the option to adjust the number.

- The first 20 short reads will be displayed on the screen.

- The first short read will be able to be highlighted using a mouse.

- Every column of characters highlighted in the first short read will be highlighted in the other 19 short reads.

- The user will highlight a column of characters that will represent the region used for BLASTing for every short read.

BLAST Stage

4.1.3.7 Setting and Changing parameters for BLAST process

Description: The system will allow users to set parameters (e, W, K, S, v and b) which affect the BLAST process. Parameters will be able to be modified. The expected failure margin, e, must have a maximum of 10. The system will also allow the user to specify the number of CPU cores(a) to use for the BLAST process.

- The system will allow the user to set the values for each parameter.

- The system will temporarily store the parameters for the BLAST process.

- The expected failure margin, e, will have a default value of 10.0.

- The window size, W, will have a default value of 0.0.

- The number of best hits from a region to keep, K, will have a default value of 0.

- The query strands to search against database, S, will have a default value of 3.

- The number of database sequences to show one-line descriptions for (V), v, will have a default value of 500.

- The number of database sequence to show alignments for (B), b, will have a default value of 250.

- The filter query sequence, F, will have a default string of T.

- The number of processors to use, a, will have a default value of 4.

4.1.3.8 Tag file generation

Description: A tag file must be generated after the user has confirmed the inputs and settings. A tag file must contain a set of unique short read and the frequency of the sequence in the input file. This will be generated by the system by counting the times a particular sequence appears in the input file.

- The system will generate the tag file after the user completes the input file stage.

- The system will allow the user to view the tag file that is generated.

4.1.3.9 FASTA formatted files

Description: The tag file and the reference files must be converted to a FASTA formatted files before the BLAST process occurs. This is because BLAST only accepts FASTA files. The tag file becomes the BLAST database, which the queries are BLASTed against.

- The family of BLAST programs converts .CSV and .TXT files to FASTA

4.1.3.10 Preview of results

Description: The first twenty lines of results will be shown on-screen to allow the user to judge whether a mistake has been made before the BLAST stage. This will be useful for Requirement 4.1.3.12.

4.1.3.11 Halt BLAST

Description: The system must give users the ability to completely halt the BLAST process. This will allow the user to make any modifications after viewing the first few results produced by the BLAST process, as explained by Requirement 4.1.3.10.

Explanation: This is required for the user to be able to return and fix mistakes they may have made.

- There will be a button that reads ’Stop’ on the screen with the BLAST output.

- When the ’Stop’ button is pressed, the entire BLAST process is stopped.

- The results up to this point will be displayed on screen.

4.1.3.12 Workload division

Description: In order to speed up the overall BLAST process, the system will allocate the workload on multiple core to do the alignment. This also means that any data that has been finished processing will be possible to be displayed, as per Requirement 4.1.4.1.

Output Stage

4.1.3.13 Tabular output

Description: The output must be in a tabular format indicating the number of hits have been found for a particular hairpin. This output is produced from the FASTA conversion generated by BLAST to a CSV file.

Explanation: This is required for the user to be able to view the results in a suitable and in a more appealing format. It will also be easier for users to access and manipulate information in files which are in a CSV format.

- The system will provide the user with an option of saving or viewing the results.

- The system will display the results in a tabular format and save the file in a CSV format.

Features

4.1.3.14 Input and reference file parsing for on screen display

Description: The system will be able to read in the files the user selects. The system must be able to correctly parse the files. The system should only read in a maximum of 20 lines for display on screen. This is due to the fact that the input and reference files are very large in size, and hence will take up to several hours to read each of them fully if we want to display the files in their entirety.

- The system will display an error message if a file cannot be read.

- The system will read in the first 20 lines of a file to be displayed on screen.

4.1.4 Optional

This section documents the functional requirements for the BLAST processing flow that are desired but considered optional by Jason. These requirements are understood to be useful to the end users, but are of lower priority compared to the essential requirements. They will only be implemented after all the essential requirements are done and if the development team has time to do so.

Output Stage

4.1.4.1 Pause BLAST

Description: The system must give users the ability to pause the BLAST process. The user will be able resume the BLAST process afterwards. This can be done multiple times.

Explanation: This is to reduce the load on the CPU if the user wants to use the computer to do a CPU intensive task urgently, as explained by Requirement 4.2.1.1.

- There will be a button that when pressed, pauses the BLAST process..

- The BLAST process will be resumed if the button is pressed once more.

4.1.4.2 Real-time update of results

Description: As threads of BLAST processes have finished processing, the on-screen results will be updated. The time between a thread's completion and the update of the results will be less than five seconds.

4.1.4.3 Saving output

Description: The user will have the option of saving the output to a file. The user will be able to specify what to name the file and where to save it. The file will be saved in CSV format and should it exceed 65,535 rows, the file should be split into multiple files.

- The user may choose to save the output.

- The user may specify where to save the file and what to name it.

- The system must be able to generate the output in CSV.

- The system should split the output into multiple files should it exceed 65,535 rows.

4.2 Non-functional Requirements

This section describes the qualities of the system, and how the system is supposed to be.

4.2.1 Essential

This section will contain essential non-functional requirements for the system that must be present in the system in order for it to be considered complete. These requirements are of high priority and must be implemented by all means.

4.2.1.1 Resource Constraint - System is not CPU intensive

Description: Simple tasks such as checking email can be done simultaneously as well as preparing for another BLAST process.

- The system will run on low priority to allow other tasks to be running at the same time.

- If necessary, the user will be able to pause the BLAST process if they have to run a CPU intensive task urgently.

4.2.1.2 Usability - Simple to use

Description: The system should be designed so that it is simple to use and easy to understand. It will be used by a variety of users with varying levels of computer knowledge. Any user should be able understand what each function does.

- Any user will be able to successfully BLAST input files against reference files in 15 minutes.

4.2.1.3 Platform Compatibility - Cross platform support

Description: The system should run on Windows XP and Mac OS X Leopard.

4.2.2 Optional

This section documents the optional non-functional requirements for the system that are desired but considered optional by Jason. These requirements are understood to be useful to the end users, but are of lower priority compared to the essential requirements. They will only be implemented after all the essential requirements are done and if the development team has time to do so.

4.2.2.1 Extensibility - Future algorithm support

Description: BLAST should be able to be swapped out for another algorithm in the future by the maintenance team.

FIVE

System Boundaries and Constraints

There should be no modifications done to BLAST. The developed software will be a GUI program which wraps around the BLAST process. No modifications or alterations should be done to the BLAST algorithm. This is a boundary the development team should not cross.

Maximum of 6GB of hard-disk storage space will be available. The 6GB includes space for the program itself and also the temporary files it would generate. This is a constraint on the development team because it will hinder our ability to deal with large files and datasets.

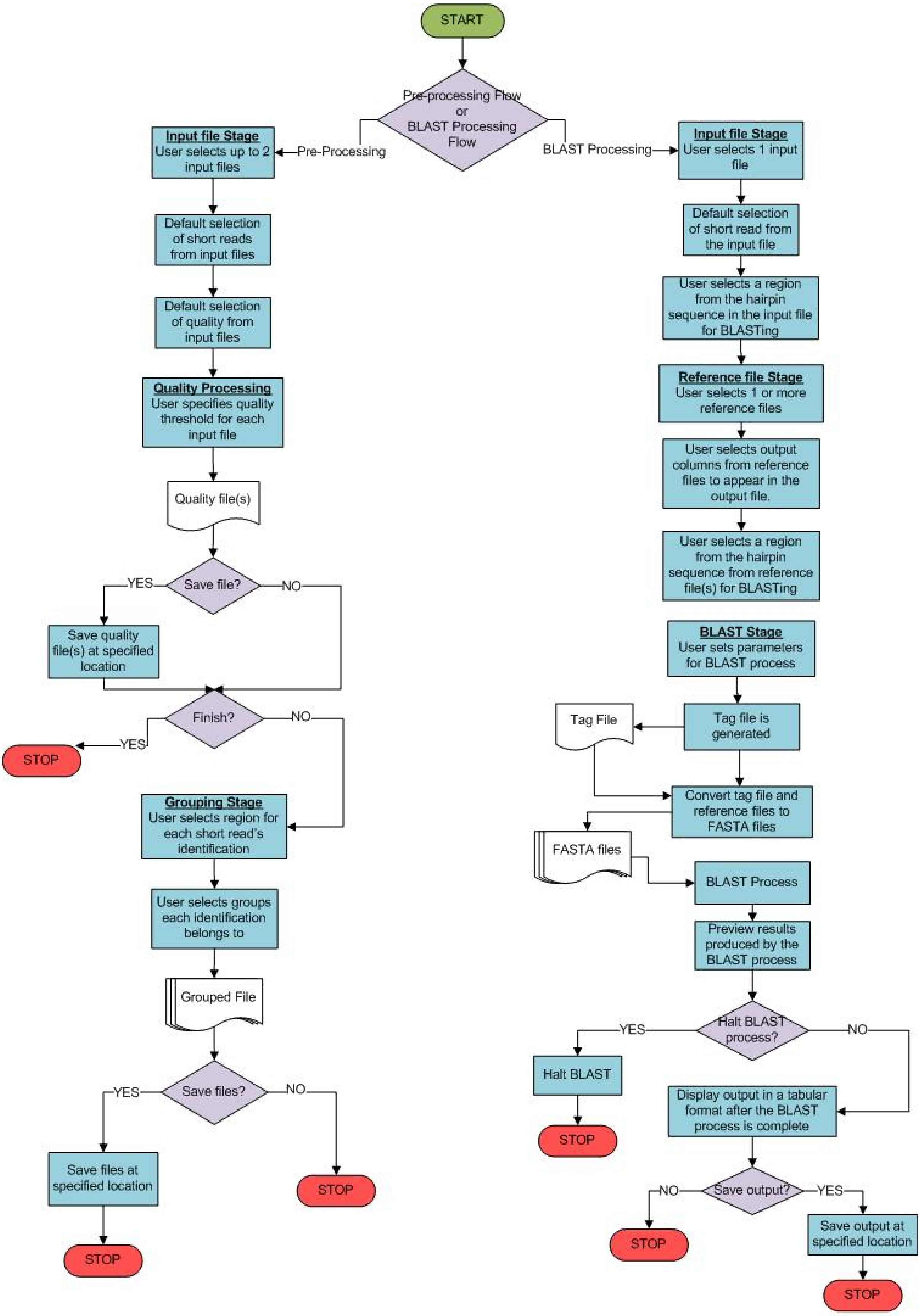
Software delivered by November There is a time constraint on the team which specifies that our software is to be deliverable by end of the semester.

SIX

Processing Flow

This section includes a flowchart to provide an overall view of the flow of the software in production.

6.1 Flowchart


START
Pre-processing Flow or BLAST Processing Flow
Pre-Processing
BLAST Processing
Input file Stage User selects up to 2 input files
Default selection of short reads from input files
Default selection of quality from input files
Quality Processing User specifies quality threshold for each input file
Quality file(s)
Save file?
YES
NO
Save quality file(s) at specified location
Finish?
YES
NO
STOP
Grouping Stage User selects region for each short read's identification
User selects groups each identification belongs to
Grouped File
Save files?
YES
NO
Save files at specified location
STOP
STOP
Input file Stage User selects 1 input file
Default selection of short read from the input file
User selects a region from the hairpin sequence in the input file for BLASTing
Reference file Stage User selects 1 or more reference files
User selects output columns from reference files to appear in the output file.
User selects a region from the hairpin sequence from reference file(s) for BLASTing
BLAST Stage User sets parameters for BLAST process
Tag file is generated
Tag File
Convert tag file and reference files to FASTA files
FASTA files
BLAST Process
Preview results produced by the BLAST process
Halt BLAST process?
YES
NO
Halt BLAST
STOP
Display output in a tabular format after the BLAST process is complete
Save output?
NO
YES
STOP
Save output at specified location
STOP

Quality file – input files that have been filtered with sufficient quality